1. **Second Highest Salary**  
   select Case when (select Count(distinct salary) from employee)<=1 then null Else(select distinct a.salary SecondHighestSalary from (select Id,salary, dense\_rank() Over( order by salary desc) sal\_rank from employee) a where a.sal\_rank=2)  
   End as 'SecondHighestSalary';
2. **CREATE FUNCTION getNthHighestSalary(N INT) RETURNS INT**

BEGIN

Set N=N;

RETURN (

# Write your MySQL query statement below.select

select distinct emp.Salary from(

select Salary, DENSE\_RANK() over(order by salary desc) as sal\_Rank

from Employee) emp where emp.sal\_Rank=N

);

END

1. **Rank Scores**  
   # Write your MySQL query statement below  
   select a.score, a.score\_rank "Rank" from (select score,dense\_rank() over(order by score desc) score\_Rank from Scores)a;
2. **Consecutive Numbers**# Write your MySQL query statement below  
   select distinct l.Num "ConsecutiveNums" from logs l, logs l1, logs l2 where l.Id=l1.Id-1 and l.Id=l2.Id-2 and l.Num=l1.Num and l.Num=l2.Num
3. **Employees Earning More Than Their Managers**select e.name "Employee" from Employee e,Employee e1 where e.ManagerId=e1.Id and e.Salary>e1.Salary;
4. **Duplicate Emails**select distinct p.email "Email" from Person p,Person p1 where p.email=p1.email and p.Id<>p1.Id;
5. **Customers who never Order**# Write your MySQL query statement below  
   select name "Customers" from Customers where Id not in (select CustomerId from Orders)
6. **Department Highest Salary**with cte as(  
   select distinct d.name as "Department",e.salary as"Salary",e.name as "Employee",max(e.salary) over(partition by e.departmentid) as "Max\_salary" from Employee e,Department d  
   where e.departmentID=d.id)  
   select department,employee,Salary from cte where salary=max\_salary;
7. **Department Top three salaries**with cte as(  
   select d.name as "Department",e.name as "employee",e.salary,   
   dense\_rank() over(partition by d.id order by salary desc) as "Ranking"  
   from employee e,department d  
   where e.departmentid=d.id)  
   select department,employee,salary   
   from cte  
   where ranking<4
8. **Trips and Users**select request\_at "Day", Round(COALESCE((Cancelled/Total),0),2) "Cancellation Rate" from( select request\_at, count(request\_at) "Total",(select count(t.request\_at) from trips t where t.client\_id not in(select users\_id from users where banned="Yes") and t.status not in ("completed") and t.driver\_id not in (select users\_id from users where banned="Yes") and t.request\_at=t1.request\_at group by t.request\_at ) "Cancelled" from trips t1 where t1.client\_id not in(select users\_id from users where banned="Yes") and t1.driver\_id not in (select users\_id from users where banned="Yes") group by t1.request\_at) a where request\_at between "2013-10-01" and "2013-10-03" ;
9. **Classes more than three students**select class from courses group by class having count(distinct student)>=5
10. **Big Countries**select name, population, area from World where area>3000000 or population>25000000 order by area;
11. **Human Traffic of Stadium**with a as (

select \*,

row\_number() over (order by id) as rn,

id-row\_number() over (order by id) as grp

from stadium

where people>=100

)

select id,

visit\_date,

people

from(

select \*,

count(\*) over (partition by grp) as duration

from a

) c

where duration>=3

1. **Rising Temperature**select w.id from weather w,weather w1 where DATEDIFF(w.recordDate, w1.recordDate) = 1 and w.Temperature>w1.Temperature
2. **Boring movies**select \* from cinema where description not in ('boring') and mod(id,2)!=0 order by rating desc